Documentation for 1d_anim_maxwell.py

This Python file simulates and visualizes electromagnetic wave propagation in a one-dimensional space using finite-difference time-domain (FDTD) method. It's a fun and engaging way to explore Maxwell's equations!

Overview

The code initializes the necessary parameters and arrays to represent the electric and magnetic fields. It then iteratively updates these fields over a specified number of time steps, producing an animation of the wave propagation.

Code Components

Imports

The code begins by importing the required libraries:

- `numpy` : For numerical operations.
- `matplotlib.pyplot` : For plotting and visualizing the data.
- `matplotlib.animation` : For creating animations of the simulation.

Global Variables

Several global variables are defined to set up the simulation parameters:

- `num_timesteps` : Total number of time steps for the simulation (400).
- `n` : Grid size, defining the resolution of the simulation (1000).
- `epsilon` : Permittivity of free space, crucial for electromagnetic calculations (8.85 × 10-12).
- `mew` : Permeability of free space (4π × 10-7).
- `Ez` : 2D array representing the electric field in the z-direction, initialized to zero.
- `Hy` : 2D array for the magnetic field in the y-direction.
- `Hx` : 2D array for the magnetic field in the x-direction.
- `c` : Speed of light calculated from the permittivity and permeability.
- `S` : Stability factor for the simulation.
- `dx` : Spatial step size (1.0 × 10-6).
- `dt` : Time step size calculated based on the stability factor, spatial step size, and speed of light.
- `lambd` : Wavelength of the wave (20.0 × dx).
- `omega` : Angular frequency of the wave.
- `Ca, Da` : Coefficients for updating the electric field.
- `Cb, Db` : Coefficients for updating the magnetic fields.

Figure Setup

A figure and axes are created for the animation:

- `fig` : The figure object for the plot.

- `ax`: The axes of the plot with specified limits.
- `line`: A line object that will be updated during the animation.
- `x`: An array representing the x-coordinates for the plot.
- `temp`: A temporary array to store electric field data for animation.

Time Loop

The core of the simulation is a loop that iterates over the defined number of time steps:

- At the first time step, the electric field `Ez` at the center of the grid is set to 1, simulating a wave source.
- The magnetic field arrays `Hy` and `Hx` are updated based on the current state of the electric field.
- The electric field `Ez` is updated based on the values of the magnetic fields.
- Every 5 time steps, the current state of the electric field is stored in the `temp` array for later visualization.

Animation Functions

Two functions are defined for the animation:

- `init()`: Initializes the line data for the animation.
- `animate(i)`: Updates the line data for each frame of the animation using stored electric field data from the `temp` array.

Animation Creation

The animation is created using the `FuncAnimation` function from `matplotlib.animation`, which will call the `animate` function at specified intervals to create a dynamic visualization of the wave propagation.

Conclusion

This code provides a simple yet powerful way to visualize electromagnetic wave propagation using Python. It's a fantastic starting point for anyone interested in numerical methods and physics simulations!